ABSTRACT

Without actually storing session-state information, the described exemplary implementations of session-state manager identify a user, validate the user's current logon state, and determine whether the user's session should expire. User identification and logon validation are checked by a server in a stateless network by generating a mathematically session-state token and sending that token to a user. Subsequently, the server receives a mathematically session-state token from the user and checks that token. If that token checks out, then the user is allowed continuing access under the same session. If it doesn't check out, then the user may be forced to start a new session by logging-on again. Alternatively, the server may check to see if the token would check out if it had come at an earlier time block. The session-state tokens are mathematical encoded and are generated using a one-way encryption scheme. Such a one-way encrypted token is scientifically impossible to reverse-engineer. Furthermore, logon expiration is checked by the server using the same mathematically session-state token. The token is checked to determine whether a predetermined number of time blocks have past. If so, then the server will terminate the user's session.